Managing the Boundary of an 'Open' Project

Siobhán O'Mahony
Harvard University
somahony@hbs.edu

Fabrizio Ferraro
IESE Business School, University of Navarra
fferraro@iese.edu

DRAFT 2.0

Prepared for the Santa Fe Institute (SFI) Workshop on
The Network Construction of Markets

Supported by Hewlett Foundation grant to
Co-Evolution of States and Markets program at SFI

John Padgett and Woody Powell, co-organizers

October 10, 2003

This research was supported by Stanford University Center for Work, Technology, and Organization, the Social Science Research Council Program on the Corporation and the Harvard University Business School Division of Research. We thank the attendees of the Santa Fe Workshop on the Network Construction of Markets for their helpful comments, in particular the helpful guidance of Woody Powell and Ben Adida. The data collection and processing efforts of John Sheridan and Vikram Vijayaraghavan were much appreciated. All errors or omissions are our own. Neither author has any financial interest in Linux or open source companies.

Abstract

In the past ten years, the boundaries between public and open science and commercial research efforts have become more porous. Scholars have thus more critically examined ways in which these two institutional regimes intersect. Large open source software projects have also attracted commercial collaborators and now struggle to develop code in an open public environment that still protects their communal boundaries. This research applies a dynamic social network approach to understand how one community managed software project, Debian, develops a membership process. We examine the project's face-to-face social network during a five-year period (1997-2001) to see how changes in the social structure affect the evolution of membership mechanisms and the determination of gatekeepers. While the amount and importance of a contributor's work increases the probability that a contributor will become a gatekeeper, those more central in the social network are more likely to become gatekeepers and influence the membership process. A greater understanding of the mechanisms open projects use to manage their boundaries has critical implications for research and knowledge producing communities operating in pluralistic, open and distributed environments.

Observers of the open source phenomena often compare the production and management of open source code to the 'open science' process of peer review (Dalle and David, 2003; Meitu, 2002; Kogut and Meitu, 2002; Raymond, 1999) where work and method are critically evaluated with informed skepticism. However, increased investment from private equity firms, attention from the press and expansion into commercial and global markets have attracted a large population of new actors interested in contributing to or selling open source software. Open source projects may desire synergistic relations with firms, and yet are wary that their culture, practice, and code may be compromised (O'Mahony, 2002) and how to maintain code quality with a growing population of contributors (Michlmayr and Hitt, 2003). Commercial collaboration has introduced new challenges: how to maintain vendor neutrality in the face of increasing industry sponsorship (O'Mahony, 2002). As open source software projects have gained greater acceptance in commercial markets, they have experienced challenges that are not unfamiliar to those faced by academic institutions.

In the past twenty years, many legal, economic and organizational scholars have become concerned that critical distinctions between public (or 'open') and private science have become blurred (Dasgupta and David, 1994). One line of research examines how the strengthening and expanding use of intellectual property protections affects future trajectories of innovation and the growth of knowledge (Lessig 2000; 2001; Scotchmer, 1991; 1999; Heller and Eisenberg, 1998). Another literature investigates how the funding, conduct, use and dissemination of university research has changed prior to and since the passing of the Bayh Dole Act (Owen-Smith, 2003; Owen-Smith and Powell, 2003; Mowery and Sampat, 2004; Mowery and Ziedonis, 2002; Mowery and Sampat, 2001) which permits an expanded role for universities in licensing their research for commercial purposes. Of

concern is whether the norms of open public science are compromised and whether enhanced private appropriation of public and private discoveries will affect the creation of new knowledge, for which the recombination and use of prior work is always essential.

Scholars in both of these areas recognize that intellectual property protections are vital to stimulating private investment in research and development. It is clear that without some assurance of the ability to appropriate rents or rewards from their investment, investors might let the most promising but risky research and development projects go unfunded. Most also agree that commercial support of university research is an imperative as public support for it has declined (Mowery and Sampat, 2004). The question is how a more integrated commercial and academic regime (Owen-Smith, 2003) can prosper without unduly influencing academic research in a direction that is not publicly accessible. These questions are just beginning to be unpacked empirically. For example, despite the fact that industry supports twice the amount of university research than it did in the 1980's, only four empirical studies of the effects of industry sponsorship of academic work were unearthed in a review of ten journals published over the past five years (Behrens and Gray, 2001).

In the next section we unpack some of the norms of open science, investigate changes in the application of intellectual property rights, and examine how this has affected the commodification of university research. With a mix of qualitative and quantitative data, we explore how one non-commercial community managed software project, the Debian project, developed a membership process to manage its boundaries and how changes in its social network affected this process.

Open Science. The logic of open science assumes that the production of science is a public as opposed to individual endeavor (David, 2003; 2001; 2000). It is characterized by norms that encourage universalistic standards based on competence or merit (Merton, 1973).

The practice of open science demands full disclosure of methods and findings, to allow scientists to replicate and verify each other's work (David, 2000; Merton 1973). This also advances a collegial community that encourages skepticism and inquiry (David, 2003; Merton, 1973). David and Dasgupta argue that the realms of science and technology are interdependent but distinct knowledge systems that diverge most in their reward systems and in their dissemination and use of that knowledge (1994).

Instead of property rights, scientists are granted priority for discoveries made. A reward system based on priority provides scientists with an incentive to share their discoveries early, thus helping to avoid duplication and advance the field more rapidly (David, 2003; David and Dasgupta, 1994). While the goal of technology is to increase the stream of rents that can accrue from their rights to private knowledge, the goal of science is to add to the stock of public knowledge (David and Dasgupta, 1994). The institutional foundations that support open science emerged in Europe in the late sixteenth and early seventeenth centuries when intellectual endeavors began to receive support form state funded academies as opposed to patronage from elites (David, 2001). Public support was crucial to fostering a climate of open intellectual inquiry without concern for property rights. However, David and his colleagues argue that the institutional framework supporting public science remains fragile and can be undermined if too great an emphasis is placed on property right protections (David, 2001).

Expansion of Intellectual Property Protections. There is some evidence that applications of intellectual property rights have expanded to new areas of academic research. U.S. patent policy has strengthened the degree of protection patents offer and extended the breadth of patents institutionally, geographically, and technologically (Jaffe, 2000). For example, the ability to patent scientific research has expanded from the expression of ideas to computer

algorithms, living organisms, and methods and processes (Coriat and Orsi, 2002). In 1979, the legendary spreadsheet software VISICALC was denied a patent (Coriat and Orsi, 2002). Now, 15% of all patents granted are for software (Bessen and Hunt, 2003). Applications for a new class of patents (705) for business methods and processes increased almost 2400% between 1995 and 2001 while the number of patents issued in this area has quadrupled (USPTO, 2001).

Bessen and Hunt's analysis of software patents shows that most are granted to manufacturers of hardware and equipment and that software patents are not correlated with a rise in investment. They conclude that software patents are more likely to be a substitute for research and development, as opposed to a complement. Bessen and Hunt argue that software patents may be used strategically to prevent competitors from making similar advances (2003). Kortum and Lerner also found that the surge in patenting since 1985 is not associated with greater research intensity, investment, or productivity, but from changes in the way people manage research and development (1998). Similar to Bessen and Hunt's finding, research investment declined slightly while patenting continued to grow. This suggests that people may be working on more applied and patentable activities or just patenting more, but not necessarily innovating or investing in innovation more.

The Commodification of University Research. A second line of research examines the institutional, regulatory and economic regime that supports University research to understand the relationships that under gird the realm of science and the realm of technology. Between 1981 and 1998, science performed at 89 Research One universities, became increasingly affected by commercial concerns, standards and rewards (Owen-Smith, 2003). Structural equation models by Owen-Smith shows that patenting experience positively affected the visibility or impact of academic work after 1983. In later years,

academic prestige had a positive affect on later patenting experience. These data suggest that in order to be successful in conducting university research, it helps to be successful in the commercialization of that research (Owen-Smith, 2003).

Further research of these same 89 US universities shows that university technology licensing offices play a critical role in assessing the commercial impact of basic research (Owen-Smith and Powell, 2003). The degree to which University technology licensing offices were connected to industry contacts affected the impact of their patent portfolios. Universities that were well connected to industry had patent portfolios with greater impact, but these relationships reached a point of diminishing returns (Owen-Smith and Powell, 2003). Technology licensing offices that were too closely tied to industry may have less innovative patent portfolios (Owen-Smith and Powell, 2003). This suggests that David and colleagues (Dalle and David, 2003; David and Dasgupta, 1994) may be right to emphasize the importance and fragility of balance between public and private interests in the production of new knowledge.

Open Source Software: New Challenges. Amidst a sharpened focus on the role of intellectual property rights in encouraging innovation and a changing regulatory climate for academic and industry collaboration, software programmers from around the world have collaborated to produce commercial grade open source software that is freely available, modifiable and distributable, and yet protected from proprietary appropriation. Software protected by licenses such as the GNU General Public License (GPL), assures potential collaborators that their efforts will not be absorbed into proprietary software (Stallman,

2003). Other works derived from GNU GPL licensed software must be guided by the same terms. This provides a common platform for collaboration[1].

While open source projects are 'open' in both process and product (source code), recent research has attended to the ways in which such projects are also bounded. For example, despite the popular belief that open source contributors give their work away, many contributors to large, successful open source projects actually own their own work and may assign copyrights to a non-profit foundation designed to hold the group's efforts in trust (O'Mahony, 2003). With growth in the scale of code, contributors, and industry sponsors, several open source projects have sought to make clearer determinations of membership and rights (von Krogh, Spaeth, and Lakhani, 2003; Michlmayr and Hitt, 2003; O'Mahony 2002). In their study of the FreeNet project, von Krogh et al discovered that potential contributors with particular 'joining scripts' and contributions of code were more likely to be awarded developer status (2003). Large successful open source projects struggle to remain open to new contributors and develop code in a pluralistic and public environment that still assures them of the source and quality of contributions received.

Unlike virtual organizations or online communities that are often described as having fluid boundaries and potentially anonymous, shifting members and identities, members of open source projects must maintain a distinct and trusted identity. This is because they are developing software that is hosted on protected servers connected to the Internet. On an open source project, the vast majority of coding and communication activities are publicly accessible and the software produced can be downloaded for free. But, access rights to the

[1] In reaction to the increased patentability of their work, some scientists and engineers have also developed sophisticated tactics to protect the areas they are working on from becoming appropriated. Such tactics include placing work in the public domain, defensive patenting, and offering rewards to research and identify p rior art to invalidate patents.

code base must be managed so as not to jeopardize its security. Project members thus do not act anonymously: establishing a distinct and respected identity is critical to becoming a member and gaining the ability to contribute directly to a project's code base. Membership can be fluid, but it cannot be *indeterminate*.

Members may not always maintain consistent activity rates, but the allocation of access rights or developer accounts must be known and distinguishable. Since contributors may never meet each other, they face a unique problem: how to verify the identities of individuals distributed around the world. Powell theorized that network forms may face novel problems of control (1990) and that membership in a community may require new organizational practices (Powell et al, 1996: 142) and, this research finds these hypotheses to be correct. Some open source and free software projects have developed a means to secure identity using public key cryptography, thus providing a unique source of network data.

Securing Identity and Project Boundaries: Cryptography and the Web of Trust. Cryptography uses mathematical algorithms to encode data so that it can travel across insecure networks and be decoded only by the intended recipients. Some cryptography methods, called secret key algorithms, use the same key to encode and decode data. This presents a complicated key distribution problem: how can a distant sender and recipient exchange this secret key to begin with? Public key cryptography solves this problem by using asymmetric keys: a public key encodes the data and a completely different private key decodes the data. This allows a sender and recipient to exchange private information without prior key distribution. A user's private key is never revealed (Network Associates, 1990). A key is merely a large number that, with the help of a particular cryptographic algorithm, like one offered by "Pretty Good Privacy" (PGP) or GnuPG (GPG), allows text to be encoded and decoded.

Asymmetric cryptography does not however solve the problem of certifying a key holder's identity. Public key cryptography secures the authenticity of the contents of the communication but it does not verify the link between the key and the sender's identity. You can prove mathematically that only the owner of the private key can perform a particular operation, but how do you prove who the owner of the private key really is? To make public-key cryptography useful, a real-world identity must be linked to a given public key. This is where digital signatures help. Public-key cryptography can be used to digitally sign documents. The private key is used to generate a signature of a given piece of data. The associated public key can then be used to verify this signature. Only the holder of the private key can perform the signature, but anyone else can verify it. Using this method, one can sign statements such as "Joe Smith's public key is XYZ" which effectively certifies the linkage between Joe Smith and his public key.

Several technical communities use the practice of 'key signing' to order to certify the link between individual identity and key ownership. A key is certified when one person digitally signs the public key and user identification packet of another. A key certification is an expression of trust: the signer believes that the public key they sign belongs to the cited person. Some form of identification documentation (usually government issued) is required to show that a public key belongs to owner and is represented by the user id packet (Brennen, 2003). This certification does not provide assurance as to the authenticity of their identification documents, but provides assurance that a particular identity is assigned to a particular key (Network Associates, 1990).

In a globally distributed environment, it will be difficult for everyone to meet everyone else. Thus, responsibility for validating public keys is delegated to trusted others. Key signers are explicitly encouraged to consider not only their own security requirements,

but, the interests of others who may rely on their judgment (Free Software Foundation, 1999).

> "Key signing has two main purposes: it permits you to detect tampering on your keyring, and it allows you to certify that a key truly belongs to the person named by a user ID on the key. Key signatures are also used in a scheme known as the web of trust to extend certification to keys not directly signed by you but signed by others you trust" (Free Software Foundation, 1999: 13)."

Certificates provide validation, but people are trusted to be judicious in the way they validate the certificates of others. A 'web of trust' is a collection of key signings that allows people to rely upon third party verification of other's public keys. The web of trust assumes that the more people who have signed each other's key (the greater the density of the network), the more reliable is the information authenticated. "The more deep and tightly inter-linked the web of trust is, the more difficult it is to defeat" (Brennen, 2003). There is no limit to the number of people that can sign a key.

One way individuals have their keys signed is by hosting or attending a 'key signing party': a get-together for the purpose of allowing people to sign each other's keys. At a key-signing party, individuals will bring a copy of their public key and valid photo identification, meet people and certify others' public keys. After a key is signed it can then be placed on a central key server that may be maintained by a 'keyring coordinator'. Key signing parties are viewed as critical to enhancing the web of trust, to teaching people about the benefits of cryptography, and to building technical communities (Brennan, 2003). They also reinforce the necessity of face-to-face contact.

> [P]lease don't sign keys of people you did not personally identify. If you don't take this process seriously, you are a weak link in the Web of Trust. If I see that you signed the key of someone who wasn't at the event, I won't sign your key, and I'll suggest that others don't either (*Key Signing Party Organizer, July 8, 2001*).

As this party organizer explains, violation of key signing protocols can lead to sanctioning and possible estrangement by other members of the group. Signing someone's key without physical verification of his or her identity breaches the norms of the community and threatens the validity of the web of trust. If someone is viewed as lax in their security requirements, then their ability to maintain the respect of their peers will be compromised.

Despite the fact that the distributed setting of an open source project implies the existence of powerful social networks, a social network approach has yet to be used to explain the evolution of its social structure. Using cryptography data from the keyring of the Debian project, we examine the evolution of its social network over a five-year period (1997 – 2001) to assess how changes in its structure affect the design of mechanisms to govern the project. Debian produces the largest and most popular non-commercial Linux operating system distribution and has been in existence for ten years. It now has over 1,000 volunteer programmers[2] distributed around the world who collectively maintain over 8,000 software packages.

The project's tenure, technical and organizational success provides a unique forum to study the emergence of their social network and their development of mechanisms to govern the project. As Figure 2 shows, the keyring network of Debian developers is characterized by a power law distribution. Most developers only have a few connections and a very small number have a very high degree of connections. To explain the growth and emergence of this scale-free network (Barabasi et al, 1999) we show how core contributors design a formalized membership process that encourages preferential attachment to gatekeepers. We explore this process of institutional design and the evolution of the social network of the

[2] Some developers engage in wage earning activities that allow them to work on Debian as part of their paid work: they are what we define as sponsored contributors. Others are volunteer. Participation in the project is always voluntary.

community. Our results show that social networks matter in the attainment of central positions in the community, but also that contributors who obtain these positions, by influencing the membership processes can enhance their advantage by reinforcing preferential attachment mechanisms. These results confirm that institutional design and social network dynamics are intertwined processes, and the former cannot be ignored when attempting to interpret the evolutionary dynamics of social networks.

Methods

The Debian project began using public key encryption as a way to build trust and authenticate member identities since 1997 and it has, since the spring of 2000, become a condition for becoming a project member. Since each key signing is dated, this data indicates when individual project members met each other. The data we collected from the Debian keyring consists of gpg and pgp keys signed by dyads between 1994 and 2001. The keyring network was only minimally active during the project's first two years (1994 -1996). Thus, we begin our analysis in 1997 when key signing started to become more widely adopted. We considered the network of Debian developers from 1997 until 2001, at the start of each year. Table 1 reports measures for the number of developers in the keyring, the rate of growth of the nodes in the network, the number of ties, average degree, S.D. of degree, number of components, and density of the network.

From the project developer database, we identified the continent of residence for each developer and gatekeeper positions, if any, held over time. In Table 2, we summarize data on the continent of residence with three dummy variables (Europe, North America, and Other) as well as other descriptive statistics. As a measure of each developer's contribution to the project, we collected data on the number of software packages each developer

maintained in 2001 and 2002, the only years available. Data on packages managed was obtained from the project's Bug Tracking Database. In both years, the median number of packages developed by any one maintainer was 4.0 and the mean was about 7.0.

Similar to prior studies of the Apache, FreeNet and GNOME projects, a small fraction of maintainers contribute the majority of the work (von Krogh et al, 2003; Mockus, Fielding and Herbsleb, 2000; 2002; Koch and Schneider, 2000). Under 8% of maintainers managed more than 20 packages in 2001 and 2002. The maximum number of packages maintained by any one person was 81 in 2001 and 101 in 2002. We also computed a measure of *package popularity*, to measure how often the packages maintained by each developer are installed and regularly used. Since early 2003, Debian users could install a "popularity-contest package" that automatically tracks and calculates statistics on the number of people that use each package regularly. We computed the raw sum of the votes for each package maintained for each developer to measure how critical each developer's contribution was for other users.

For each year we computed a measure of developer project tenure, counting the months since they first signed a key. Betweenness centrality is a measure that synthetically captures the structural position of developers in the social network and each individual's ability to potentially broker information and exert social influence. In this context, betweenness centrality is a measure of an individual's ability to link disconnected parts of the network through face-to-face interaction[3]. Betweenness centrality (Freeman, 1979; Marsden, 1982; Wasserman and Faust, 1994) measures the extent to which an actor can broker communication between other actors. The index can be computed as follows:

[3] We also computed a measure of degree centrality, simply measuring the number of ties each developer had, but since this measure was highly correlated with betweenness centrality, we only used the latter in our analysis.

$$C_B(n_i) = \frac{\sum_{j<k} \frac{g_{jk}(n_i)}{g_{jk}}}{\frac{(g-1)(g-2)}{2}}$$

Where g_{jk} is the number of geodesics linking two actors, $g_{jk}(n_i)$ the number of geodesics linking two actors that contain actor I, and $g_{jk}(n_i)/g_{jk}$ the probability that actor is involved in the communication between two actors. By standardizing its value by the number of pairs of actors not including n_i (g-1)(g-2)/2, the index will take values from 0 to 1.

In order to understand this data in the context of the project's evolution, 76 informants from the Debian user and open source community at large were interviewed, six of them in leadership positions within Debian. Online documentation such as mailing list archives, meeting notes, and other formal project documents offered an additional source of data. In addition, key events that took place in the open source community and the computing industry at large were identified through Internet searches.

We used Pajek software (Batageli and Mrvar, 1998)[4] to generate a replicable visualization of the network of Debian developers. The drawings were constructed by using the algorithm Kamada-Kawai, which locates connected nodes adjacent to one another and set the distance between them as a function of the shortest network path between them. The distances between unconnected portions of the network are undefined in this algorithm, and therefore the relative position of different components does not have substantive meaning (Kamada-Kawai, 1989). Vertices were scaled to represent the number of packages a developer maintains for the project, and their colors indicate either their geographic locations (in Figures 3-8) or whether the developer was a member of the New Maintainer

[4] This software, developed by Vladimir Batagelj and Andrej Mrvar, is freely available for noncommercial use, and can be downloaded at http://vlado.f.f.uni-lj.si/pub/networks/pajek/. See de Nooy, Mrvar, and Batageli (2003) for an introductory text to exploratory social network analysis with Pajek.

Committee (in Figures 9-10). The New Maintainer Committee (NMC) was created in 1999 to design a new developer admission policy. To show how the structural position of the developers affected their ability to become members of the New Maintainer Committee we estimated a logit model (Long, 1997). We tested whether centrality in the network had any impact on this outcome in the period 2001-2002, controlling for level and criticality of contribution, tenure in the project and geographic location.

The Evolution of Debian

The Debian project was initiated August 16, 1993 with an announcement to a Usenet newsgroup. The project founder proposed developing a non-commercial, easily installable packaged version of the GNU[5]/Linux operating system that would be managed differently from the Linux kernel. The stated goal was to create a complete operating system that would be 'commercial grade' but not, itself, commercial.

> Rather than being developed by one isolated individual or group, as other distributions of Linux have been in the past,[6] Debian is being developed openly in the spirit of Linux and GNU. The primary purpose of the Debian project is to finally create a distribution that lives up to the Linux name […]. It is also an attempt to create a non-commercial distribution that will be able to effectively compete in the commercial market (Murdock, 1994).

About two-dozen people responded to the initial Usenet posting and the founder created a new mailing list specific for this project named "Debian."[7]

[5]GNU is a recursive acronym that represents the phrase "GNU is Not Unix". The GNU system developed by Richard Stallman was designed in opposition to the proprietary restrictions associated with UNIX.

[6] This reference to other Linux distributions managed by one person likely refers to the Linux kernel managed by Linus Torvalds.

[7] The origins of the project's name stems from a combination of the founder and the founder's wife's names.

Between 1993 and 1996, the founder, with the help of Usenet respondents, collectively designed a modular package management system. A package is a unit of code that can be maintained independently from the rest of the operating system but has a standardized interface that allows integration with other packages. To maintain a package is to manage the receipt and review of code contributions from other contributors (called 'upstream maintainers') and 'package' these smaller contributions into a discrete module. A modular package system enables many people who are not physically co-located to contribute to the project by permitting different development activities to be conducted in parallel. From 1994 to 1995, the Free Software Foundation supported the founder to design a technical infrastructure that could handle multilateral contributions. The first whole number release (1.1), announced in June of 1996, had 474 packages.

Initiating the Web of Trust. By January of 1997, nine members from the United Kingdom connect and two isolate pairs, one in Germany and in Sweden are formed (See Figure 3). By June, 41 contributors joined, 73% of them from Europe. Neither the founder (who resigned at the end of 1996) nor his subsequent appointee (who presided throughout most of 1997) appear in the network. The most central developer, who was one of the first keysigners in 1994 would become the project's third leader in 1998.

In the months leading up to the July 1997 release, members debated how to manage distribution of the release and the project's non-commercial status. Five issues were prevalent on their mailing list: 1) how to garner commercial legitimacy for the project; 2) how to logistically distribute their software; 3) how to raise funds to support the project's legal expenses; 4) how to distinguish 'official' copies of Debian from versions modified for commercial purposes; and 5) how strongly the project should encourage firms to contribute to the project.

Exploration of some of these concerns challenged the meaning of 'non-commercial' as it was initially conceived. Other concerns were more pragmatic and reflected competing goals unequally shared by members: to both control their product and yet disseminate it broadly.

> We don't want to be in the CD manufacturing business, the import-export business, or the order fulfillment business. We want to get Debian into as many people's hands as we can, for as little money as possible (*Posting to Debian Development Mail list, January 17, 1997*).

Project members wanted to acquire the legitimacy associated with shrink wrap software, but Debian did not have the capital to manufacture a physical distribution. Commercial involvement would help them establish a larger market share than Internet downloads would permit, but they did not want to sell their work.

One proposal to contract with firms to distribute Debian for two dollars was perceived by others as crossing the 'non-commercial' line. Project members struggled with whether it was within their charter to ask, mandate, or suggest contributions from firms. The individual that proposed the fee angrily defended his idea to improve the commercial appeal of the project in the following post.

> I AM NOT TRYING TO TURN THE PROJECT INTO A COMMERCIAL ORGANIZATION. IT IS A NON-PROFIT. I WANT TO RAISE OUR PERCEPTION IN THE PUBLIC BY MAKING OUR PRODUCT _LOOK_ COMMERCIAL (*Posting to Debian Development Mail list, January 19, 1997, original format*).

What a non-commercial distribution could be was hotly contested and in the end, a consensus agreed that Debian would not sell. Individuals and firms could freely download and resell the Debian distribution with no fee. In return, some firms made free copies available to Linux User Groups or donated a portion of their proceeds to Debian. In July,

the second project leader announced the first 'official' release (1.3) with 974 packages contributed by 200 developers.

Success and New Vulnerabilities. The keyring network grew 531% in 1998 and 114% in 1999, to 176 nodes (Figures 4 and 5). This rapid growth in contributors was likely stimulated by commercial interest and media attention (Figure 1), but growth in the keyring itself may also have reflected a growing concern over the threat of 'Trojan' contributors. A Trojan contributor was a volunteer or 'malicious contributor' who purposively introduced bugs or viruses to the project. Debian's growing success and popularity meant that, like other Linux distributions, it could be considered a threat to developers of other commercial operating systems. However, well-intentioned but unskilled developers could create equally detrimental effects. Debian developers all have the same access rights, and can upload anything into the project's code archive, which has the potential to affect all other packages. However, changes made by a non-maintainer will not carry the same status as those made by someone listed as the maintainer[8]. If someone fixes a bug in someone else's package, that bug will be tagged and fixed, but the maintainer will have to close it, signaling that the person responsible has reviewed the work of the non-maintainer. Newcomers to Debian who were not fully cognizant of Debian procedures could wreak havoc. Members were torn between reconciling the need to welcome people interested in Debian with the need to protect the project from potentially destructive outsiders.

Mailing list archives indicate that the idea of using a key ring to authenticate contributor identity was proposed by the person who initiated the key ring network in 1994.

> I think that at least one of our objectives should be to establish a socio-legal comeback in the case of a malicious developer. This means that we need to verify the real-world identity of the developer somehow. There are several ways to do this,

[8] See Michlmayr and Hill (2003) for more description of the new maintainer upload process.

> including personal introduction by an existing developer, commercial key-signing, attempting to use PGP web of trust, telephone verification of some kind (*Posting to Debian Development mail list, February 28, 1997*).

At the time, there was no formal standard membership process and little preliminary screening. As the informant below describes, the ability to articulate areas for contribution was considered evidence of one's capability to work on the project.

> When I applied, I told [the Debian Project Leader], "Here are the packages I want to work on." Back then it was pretty easy and we didn't do the identity check at that point. It was pretty easy to assume that if you knew about Debian back then, you were fairly competent and probably understood the basics of what free software was about. It was a new thing back then — free software particularly. (*Sponsored Contributor, Former Volunteer, November 9, 2000)*

Several mailing list threads discussed ways to secure the identities of contributors, ascertain membership, and determine project decision rights. "[P]erhaps a distribution like Debian needs to have a registered list of users who want to vote on policy matters. There are enough psychos out there to make sure that groups such [as] Debian do not succeed" (*Posting to Debian Development Mailing list, October 25, 1997*). This discussion would persist for some years. How could they keep the project 'open', but ensure that contributors were not only well intentioned, but skilled enough not to inadvertently harm the project? Members were reluctant to articulate a formal set of skill requirements or make too many demands of volunteers. "I'm not sure about competence or integrity requirements [to acquire maintainer status]; somehow it goes against the grain for someone who is not issuing my paycheck" (*Posting to Debian Development Mail list February 27, 1997*).

The second project leader stepped down in November of that year and the initiator of the key ring network became the next leader in a de facto unopposed election. The third leader did not have the highest centrality in the network, but occupied a brokering role

between the largely American component of the network and the more densely connected European component (See Figure 5). The second whole number release (2.0) was announced in July of 1998 with 1500 packages and over 400 developers. As Figure 1 shows, this release coincides with a sharp increase in media attention devoted to Debian. Shortly after, Debian's fifth birthday was celebrated with an IRC party in August.

The third leader initiated the collective drafting of a Constitution, ratified by 86 developers, that outlines the roles and rights of the Debian Project Leader (DPL) and project members. The DPL has power to make decisions for which no one else has responsibility but is directed to "avoid overemphasizing their own point of view when making decisions in their capacity as leader" (Debian Constitution, Article V). Members differed on what they thought a DPL should do and over the years DPLs varied in the degree to which they focused on internal coordination of the project.

> It's not clear exactly what [a] DPL is supposed to do. Like what do people expect? Because Debian is so big, everyone expects something else. So, some people say, "Yeah the DPLs should only go to conferences, and present Debian to the outside world, but he shouldn't do anything internal because that is working any way." Whereas I think it's very important to do coordination in the project in multi-ways and things like that. (*DPL, June 20, 2003*)

The DPL appoints a Project Secretary and Technical Committee empowered to "decide any technical matter where Developers' jurisdictions overlap" or where efforts to resolve a decision have failed (Debian Constitution, Article VI). A supermajority (3:1) is required to overrule a developer.

In addition to delimiting the authority of Debian leaders, the Constitution also bounds the group's authority over each other. Members have the right to: 1) Make technical or non-technical decisions with regard to their own work; 2) Propose or sponsor draft resolutions; 3) Run as a project leader candidate in elections; and 4) Vote for resolutions and

leadership elections. When Debian tested its new Constitution with its first official election, 60% of voters helped usher in the project's fourth leader. The Constitution details specific privileges for developer members, but does not articulate how one becomes a member. The question of how to prevent a 'Trojan' member was left unresolved as Debian's public presence continued to grow.

Membership Crisis. In August of 1999, several package maintainers who were not yet granted developer status began complaining of the wait to obtain a developer account. The Developer Accounts Manager (DAM), authorized to assign new accounts, faced a backlog of people interested in the project and had no real way to ascertain the qualifications of a particular maintainer. A few resigned in frustration. A contributor that proved their ability to maintain a package could become a maintainer, but they did not necessarily become a developer or project member. Only developers had accounts to access the code repository and in order to upload a package directly, they had to sign the package with their key.

> If anyone could upload to Debian… [expression]. We had to guarantee that it actually comes from a trusted source, and that it hasn't been changed along the way. That is how what we achieve this - by assigning the packages….First of all it shows that it is from a trusted source. It is signed by a key, which is a developer. The other thing is it shows that the package hasn't been modified. Like during the upload someone could come and change the package perhaps. (*Volunteer Contributor, DPL, June 20, 2003*)

To avoid losing contributors and their work, some developers began 'sponsoring' member candidates by uploading their packages for them and signing them with their keys. Postings to the list and interviews with informants suggest that the delay in accepting new maintainers reflected a heavy workload for volunteers, but also their concerns that recent entrants to the project were actually hindering the project more than they were helping.

> The problem is they [new maintainers] are not contributing. And some of them are contributing bugs. They are actually adding bad packages. And there was a lot of kicking from old developers about the new people and how they are not reading anything and doing things, it is always old versus new. So we closed being a new maintainer for while because there were so many new packages coming in that they were not helping with anybody else's packaging, they were just uploading their own. So what we were seeing is 1,000 new packages a year and that many more bugs per package showing up. But no change right? It was not getting better, it was getting worse. So we closed it off for a year. Then we sat down and wrote up how we wanted it to work (*Sponsored Contributor, March 20, 2001*).

After doubling in size, project members were frustrated by the growth in bugs relative to contributions from new members, and in particular, the age of bugs that remained unaddressed. New maintainers wanted to work on areas of their interest, not debug existing bugs.

In response to complaints about the long delay to become a developer and to address concerns about accepting unqualified members, the DPL made a controversial move: he closed the project to new maintainers until a new membership process could be designed. Membership would remain closed until April the following year: almost six months. As Figure 1 shows, Debian's contributor and package growth plateaus slightly.

> Debian's new maintainer team is currently not processing requests. The team wanted to resolve some problems they observed with the way Debian maintainership is currently handled, and decided to close new-maintainer until these have been fixed. We are currently working on a new structure for handling new-maintainer requests, and hope to have this finished as soon as possible (*Debian Project Leader, Posting to Mail list October 11, 1999*).

The DPL followed this announcement with a recruiting call outlining criteria for the New Maintainer Committee (NMC). Developers were invited to email the leader (privately) and were told that committee members would be selected according to the following criteria:

> [T]he following guidelines will be used in selecting new members to the new-maintainer team:

> - needs to have a *strong* opinion for free software
> - he needs to be able+willing to make long distance phone calls
> - He needs to know what he's doing, that new people need some guidance, we have to prevent ourselves from trojans etc.
> - we need to trust him - more than we trust *any* other active person
> - He *has to* understand that new-maintainer is *more* than just creating dumb accounts on n machines (*New Maintainer Proposal, October 19, 1999*)

The need to "trust committee members more than any other active person" suggests that the DPL understood that members of this committee would become future gatekeepers for the project and wanted to have confidence in their philosophical commitment above and beyond their degree of effort on the project.

Committee candidates were asked to uphold the ideals of free software, help newcomers, and detect or prevent 'Trojans' or nefarious outsiders. The committee's first proposed membership process had four-stages: initial contact (with possible phone interview), checking identification, internship, and acceptance. While there had always been some identification process before granting new developer accounts, it was not standardized. The second stage would ensure that members "know that the person actually exists as the person that they say they are, that there is a known location for that person where they can be spoken to, the person's current situation, as well as [the person's] long term contact information must be clear" (New Maintainer Proposal, October 17, 1999).

One member, questioning the need for a phone interview, suggested that face to face meetings would help instill greater collegiality, trust and respect among project members and foster more keyring signing.

> Maybe the "meet a developer approach" combined with a brief phone interview is better than a lengthy call from some faceless developer. Plus it gives new maintainers an opportunity to have their key signed, which helps build our web of trust, and the personal contact might socialize against flamemongering (I suspect I'll be better behaved on the lists now I've met a few of you at ALS, for example ;) (*Posting to Debian Project Mail list, October 18, 1999*).

The biggest suggested change was an 'internship period' that would "allow the applicant to prove himself". The internship would target four concerns: technical competence, knowledge of organizational procedures, collegiality and commitment, and philosophical agreement with the principles of the project.

> It [the internship] allows us a good method to help a new maintainer with his new work and teach him about the Debian system (both technical and organizational). It allows us to get to know the person: is he responsive to bug reports or other requests, is he able to produce a quality product, and also very important: does he agree with our philosophy? (*New Maintainer Proposal, October 19, 1999*).

Were these requirements too onerous for an open project that valued freedom?

This proposal led to a discussion of what it meant to be an 'open project' with a Constitution that places no restrictions on its members. Some wondered whether the DPL had exceeded his authority or whether a Constitutional amendment would be required.

> What are the reasons for ever not letting new maintainers in? There are none, I agree. I'm very disappointed that [DPL #4] has failed to reopen New Maintainer. This is the biggest failure of his tenure thus far, IMHO (*Posting to Debian Project Mail list, December 29, 1999*).

Ready and willing would be contributors posted their frustrations with the closed process and the lack of clear criteria for membership.

> My understanding is that the addition of new maintainers is not merely slow, but has been officially stopped. Why? What IS the motivation? Here I am, a highly competent person, a happy and satisfied Debian user, and someone who thinks it's my duty to contribute back to Debian with some of my labor and talent (*Debian Project Mail List Posting, December 20, 1999*).

However, the new maintainer process did not reopen before the year's end.

Taking Leadership and Designing the Membership Process. The New Maintainer Committee (NMC) was poised to modify the future structure of the social network by developing a

process that would regulate the flow of new members. Who would become gatekeepers of the project? Table 3 presents the logit coefficients for models predicting membership in the NMC from the number and popularity of packages maintained, tenure, geographic location and brokerage position of developers in 2001 and 2002. In the base model, which does not include network centrality, only the number and popularity of packages maintained have a positive significant effect on the likelihood of joining the NMC in 2001, while tenure has a negative effect. In the full model for 2001, developers who contributed more packages, worked on more popular packages, had shorter tenure on the project, and occupied brokerage positions in the network, were more likely to become members of the NMC.

Similar to Fleming and Waguespack's study of the emergence of leaders in a technical community, technical contributions are predictive up to a point, but degree has a significant and positive influence on leadership (2003). In their study of leadership on the Internet Engineering Task Force (IETF), the 'reflected status' of others (as measured by publications of their colleagues) also predicted leadership (Fleming andWaguespack, 2003). One might view keyring signing as another form of 'reflected status', a sign of the number of people that vouch for your identity. While effort is predictive in our study, occupying a central position in the network was more predictive of membership on the NMC. Maintaining one more package increased the likelihood of becoming a member of the NMC by 4%[9]. With a one percent increase in betweenness centrality, a developer is three times more likely to become a member of the NMC.

[9] To help the interpretation of the logit coefficients, the odds ratios are reported in parentheses. Odds ratio are computed by taking the antilogarithm of the logit coefficient, thus for the effect of the number of packages in 2001, we can simply compute: $e^{04}=1.04$. Values exceeding 1 indicate and increased likelihood of becoming a member of the NMC, while value less than 1 indicate decreased odds.

The popularity of one's package is also predictive of NMC status. For every 100 people who use a developer's package, he or she is 4% more likely to become a NMT member (3% in 2003). Tenure likely had a negative effect because those who joined the project more recently were more likely to be aware of the problems with admitting new members. In 2002, these results are confirmed, even though the magnitude of the effect of centrality is smaller (Odds ratio=1.47).

The average degree of developers, stable from 1997-1999, increased to 3.64 in 2000 with a S.D. of 4.67. This was much higher than the S.D. in the two previous years (Table 1). After 2000, the average degree of the network and its S.D. increase every year. The increasing spread of degrees per developer could be affected by newcomers' preferential attachment to members of the NMC. That is, if centrality affected the likelihood of becoming on the membership committee and the new membership process required face-to-face contact, then new members would preferentially attach to NMC members (Barabasi et al, 1999). Those on the committee would be more likely to continue to meet more people cultivating accumulative advantage (Owen-Smith, 2003; Merton, 1968). Merton describes the structural effects of accumulative advantage in the words of St. Matthew: "For unto everyone that hath shall be given, and he shall have more abundance" (1968:58). Merton used this concept to explain the continuation of recognition and reward for proven scientists and the difficulty unproven scientists had in accumulating recognition. In our context, accumulative advantage accrues to those who are able to meet more people and become central to the network. They will help guide future determinants of network expansion.

Indeed, the new membership process designed by the NMC required not only identity verification through face-to-face exchange of keys, but sponsorship by an existing member, demonstrated understanding of the community's philosophy and procedures;

demonstrated technical capability; and a written recommendation from an Application Manager. The first new Debian member formally admitted since October, 1999 was admitted in April, 2000. By November 10, 2000, a 100 people had passed the new maintainer process and several hundred were in progress.

Debian did not have the resources to physically bring all project members together, but other events helped grow the keyring network. About 30 developers attended the first formal organized face-to-face meeting held in the summer in France. At a Linux tradeshow, the DPL received a cash award on behalf of the Debian project. Such awards grew the coffers of Debian's non-profit and enabled them to occasionally sponsor a trade show booth or support a developer's travel. Linux's technical success in the marketplace meant that there were more firms likely to hire programmers to work on Linux, Debian or other open source projects and sponsor their travel to conferences and tradeshows. Unfortunately, data on firm sponsored contributors to Debian is not available, but informants reported that greater support for travel was one positive side effect of the commercialization of Linux. This could provide another possible accumulative advantage feedback loop. If sponsored contributors were more likely to travel and meet other developers, than this could affect the degree to which they became central in the network, which would then has a positive affect on acquiring an NMC or leadership position.

In 2001, the new maintainer process would mature and become stricter yet as the network doubled in size. By the beginning of 2001, the keyring network had 532 nodes with 1212 ties. The frequency distribution of degrees provides some evidence of an accumulative advantage effect (Figure 2). Forty-three percent of developers in the network had met one another developer, while very few had met a large number of other developers. In March, the fifth Debian leader was elected with 311 unique votes by the closet margin on record.

The runner up candidate articulated many of the difficulties that underlay the new maintainer membership process and argued that they were the product of Debian's unusual success.

> The biggest frustrations I see with Debian are, in fact, all related to this success. We have more developers than ever, more packages soaking more bandwidth to mirror than ever, more open bugs than ever, and our user community is broadening into areas where the criteria for success may be different. This puts enormous pressure on our organization, forcing us to continue to evolve…. (*DPL Candidate Platform, February 23, 2001*)

This candidate, in the top 5% of maintainers in terms of the number of packages managed, did not win the election but ran again and won in 2002.

Narrowing the Pipeline. At the top of the fifth DPL's list of proposals for Debian was "adding more structure to the project", namely to improve the new maintainer process. He noted that the NMC was under incredible pressure to protect the Debian project from 'Trojans' and that the current process was still not robust enough to handle further growth.

> [W]e all have the same permissions to upload packages, we all have just as much right to screw up the archive as anyone else. In there lies the problem. Maintainer count is on the rise all the time. It may not seem to be a problem now, but increasing administrative and security work to keep this increase on a good footing is not going to remain easy (well, it isn't easy now). Stopping developer entrance all together is not an alternative either. Some may argue that this makes Debian less "open". Well, I don't think Debian is closed at all…..Now I know this isn't the same as having one's own packages, and that sponsorship is not turning out to be the godsend that we had hoped. However, allowing more levels of maintainship will likely make it easier for people to contribute. (*Leadership Platform, February 20, 2001*)

The DPL's approach to keeping Debian open, but managing its boundaries, was to create differentiated levels of access to the code base. This change did not get implemented, but the process did become stricter.

When the new maintainer process was adopted in the spring of 2000, there was a backlog of applicants and the 'Front Desk' position worked diligently to quickly assign

candidates to Application Managers. After while, Application Managers found that many applicants were no longer interested or responsive. The head of the NMC proposed narrowing the pipeline by requiring applicants to obtain a sponsor before applying.

> An increasing number of applicants are either not serious about joining Debian and contributing to the project or not well prepared for the new maintainer process yet. A proposal is made to require all potential developers to maintain the recommendation of an existing developer…Those not very serious in joining are thus not able to apply in the first place (*Changes to the New Maintainer System, February 5, 2001).*

This change, accepted as a means for members of the NMC to save time, effectively eliminated the possibility of newcomers without prior attachment to a network incumbent, further reinforcing the potential for accumulative advantage. Some members felt that the approval requirements were too onerous and undermined the freedoms Debian espoused.

> Raising the threshold too high, or even just the perceived notion, whether justified or not, that many NMs [New Maintainers] are unskilled, could make Debian more and more like an elitist society……As for me, I am just glad that I became a Debian developer over 3 years ago, long before this was even an issue. (*Posting to Debian Project, January 7, 2001*)

Even the current DPL echoed the relief expressed by the informant above. He agreed that it was much harder to become a member now than when they had started.

> At the time new maintainer was not as formal or anything as it is now. For example if I look at my reports, my [Application Manager] reports were excellent at the time. …If I look at them now they are crap compared to today's standards, because we are asking many more questions, and we're doing more checks and everything. It is much more complicated now. (*2003 DPL, June 20, 2003*)

As of this writing, 567 members have gone through the new process, 158 were in progress, 36 were awaiting a sponsor, and 18 are on hold. Thirty-five people in 17 countries[10] were looking to have their GPG key signed and almost 200 people in 28 countries offered to sign keys.

In 2001, Debian was regarded by the press as technically sophisticated, but lacking the authenticity and reliability of a company. When reviewing Release 2.2, the editors of PC Magazine reported that "From a corporate standpoint, Debian's main drawback is the lack of a company to support it…Debian's developer community is very active, open and approachable, and Usenet groups and mailing lists are abundant, but don't expect a lot of novice-level explanations. Commercial technical support for Debian is only available through third parties" (Ulrich, 2001). Debian's informal and loose organization permits qualified developers to work on packages of their choosing, conditioned only by what their colleagues are doing. The project remains non-commercial and their development environment and source code are publicly accessible. However, becoming a developer is no longer as easy as it once was. It can now take six months or more for a new maintainer to become a developer. In fact, it may be easier to join a firm. The new membership process requires preferential attachment: creating a tie to a developer who most likely already has a large number of ties.

Discussion

The social network of Debian developers, as captured by the keyring data, has a pragmatic purpose: identity verification. The growing popularity of Linux in general and

[10] These countries included Argentina, Belgium, Brazil, Canada, Germany, Denmark, Spain, France, United Kingdom, Hungary, Indonesia, Mexico, Netherlands, Norway, Philippines, Singapore and the United States.

Debian Linux in particular created a practical problem that may be present in other types of distributed knowledge producing communities (David and Foray, 2002). Without a central sponsoring organization, how could Debian protect against "trojans" and maintain an open environment that was conducive to the norms and values of the open source community? The explosive unregulated growth of developers, partially due to the media attention, and partially to the technical and market success of the product, led developers to fine tune a new social structure that could help ensure that new member's skills, goals, and ideology were in line with that of the collective.

Despite the fact that some researchers have argued that open source projects do not rely on trust (Gallivan, 2001), prior research has shown that trust is critical for virtual teams (Jarvenpaa and Leidner, 1999) and this case is no exception. As Nohria and Eccles predicted (1992), "there may be minimum ratio of face to face to electronically mediated exchange that is vital to maintain in order for a network organizations to work effectively" (1992: 290). A crisis occurred when the project no longer felt that it could adequately protect its boundaries and closed its doors to new potential members. This led to the constitution of the NMC and the articulation of membership criteria and a process, thereby institutionalizing the key ring.

As Barabasi and colleagues explain (1999; Barabasi and Albert, 1999), network expansion and preferential attachment alone can produce a scale invariant distribution. Networks expand with the addition of new nodes. If new nodes are more likely to attach to nodes that already have a large number of connections, then the network will follow a scale-free power-law distribution (Barabasi and Albert, 1999). In networks with a power-law tail, there is a high probability that a small number of nodes will be highly connected. As Figure 2 chronicles, from 1997 to 2001, the keyring network increasingly conformed to the power-

law distribution. While Barabasi and Albert argue that these effects can be found in natural, technical, and social systems (1999), how such processes are enacted at the micro-level in social systems is unclear.

In this research, we showed that meeting people face-to-face and acquiring a central position in the network enhanced the probability of attaining a gatekeeper position far more than the number of packages maintained. Our qualitative data shows that once people joined the New Maintainer Committee, gatekeepers determined the new rules for membership and designed a membership process that required sponsorship and a face-to-face meeting. Network visualization techniques illustrate how a scale-free network becomes connected over time. This combination of research approaches illustrates how processes of accumulative advantage can affect the structure of the network as well as the design of governance mechanisms.

The institutionalization of a membership authentication process creates a new social network and in doing so, influences gatekeeper selection and future expansion of the network. Attaining a gatekeeping position in Debian is now consistently associated with occupying central positions in the network, even after controlling for technical contribution, tenure and geographic location. This is despite the fact that most researchers and contributors to open source and community managed projects claim that such projects are guided by purely meritocratic concerns. While developers coordinate their work almost solely online, they seem to trust people they have met more when they have to choose individuals to manage their boundaries. The new maintainer process, with its requirement of sponsorship, contributes to the centralization of the network. Although we do not have the data to run a Logit regression for 2003, we were not surprised to learn that the head of the New Maintainer Committee was elected Project Leader in 2003.

This research also shows how a reinforcing cycle of accumulative advantage helps sustain a scale invariant distribution. Previous research on the evolution of networks in the biotechnology industry (Powell et al, 2002; Owen-Smith et al, 2003) found that actors in this field had a preference for diversity as opposed to incumbents or like others (homophily). The recombinative nature of innovation in the biotechnology field demands a constant quest for new partners and ideas (Powell et al, 2002). However Owen-Smith et al (2003) find that this preference for newcomers is modified. An 'open' elite, that is receptive to network entrants but yet defines the standards of action for the field, emerges.

Similarly, in this analysis of Debian's social network and new governance mechanisms, the elites that emerge define the terms of admittance. In doing so, they help the project to stay open and enhance their own structural advantage. Barabasi and Albert suggested that scale-free networks "are the inevitable consequence of self-organization due to the local decisions made by the individual vertices, based on information that is biased toward the more visible (richer) vertices, irrespective of the nature and origin of this visibility" (1999: 512). We argue that in many real world social networks such bias can be manipulated by design. Here we provide some insight as to the origin and nature of structural advantage in a social system that exists primarily on-line, but requires real world authentication.

This research has broader implications for far-flung research and knowledge communities that rely on emergent and novel governance mechanisms. First it suggests that literature on virtual communities and new organizational forms remains incredibly naïve to the realities and challenges of an online production community. New organizational forms and virtual organizations are often depicted as nebulous and constituted by a shifting and ever changing body of members. Volatility and turnover in participation in new

organizational forms is not the same as determinacy. Organizational theorists who use terms such as porous and permeable organizational boundaries need to be more precise in this regard. Flow is not the same as ambiguity. Boundary definition and management is likely to continue to be critical for research and other knowledge producing communities that wish to be open, but must do so without jeopardizing the security and stability of their work.

> What is at stake here is the entire range of mechanisms that will facilitate interpersonal and inter-organizational transactions, given the new conditions for knowledge transactions and exchanges: increasing specialization, increasingly asymmetrical distribution of information and assessment capabilities, ever-greater anonymity among interlocutors and ever-more opportunities for forgery of identity. Clearly new methods need to be devised to "Certify" the knowledge circulating on the Internet….. (David and Foray, 2002: 17)

If distributed, pluralistic knowledge producing communities that cross or exist outside of established institutional boundaries are to survive, innovations in trust mechanisms will likely continue to emerge. More research on mechanisms that help carve a protected and common informational space for collaboration is needed to understand not only their emergence, but their contribution to project security and effectiveness.

Such research should attend to the legal platform required for such work to flourish as well as the socio-technical space for collaboration. Prior to reaching the point where growth created membership and boundary challenges for Debian, all potential contributors were assured that the work created by Debian members would remain non-proprietary and that the Debian project was committed to being non-commercial[11]. While not the focus of this paper, intellectual property concerns remain a constant source of concern for the Debian project. New challenges with regards to derivative works, proprietary modules and

[11] For more information on Debian's Social Contract with its members and community of users, refer to: http://www.debian.org/social_contract or O'Mahony (2004) for further analysis.

drivers often inspire divisive wedges among members as to where the non-commercial boundary of Debian falls.

It is argued that in response to challenges to the norms of open science, well-defined communities of academic practice and credit are giving way to much larger co-owned intellectual property resource pools (Hellstrom, 2003). If academic research and commercial research are forging new areas of interdependence, particularly in the biomedical field, then scooping rights of access and belonging to emerging research opportunities will be a future challenge. David and Foray acknowledge as much in their prescient analysis of knowledge communities:

> The potential for producing and reproducing knowledge will become greater as a community expands, but then so will the costs of data search, the risk of congestion and anonymity amongst members, which can in turn, represent a source of acute problems of trust. (David and Foray, 2002: 8)

As illustrated here, these challenges are very real and the design of new mechanisms to manage them will co-evolve with the social networks that underlie such communities. The power of such mechanisms to affect the future trajectory of the project should not be underestimated.

References

Barabasi, Albert-Laszlo and Reka Albert. (1999). Emergence of Scaling in Random Networks, *Science* 286 (509-512).

Barabasi, Albert-Laszlo, Reka Albert, and Hawoong Jeong.(1999). Mean-field theory for scale-free random networks, *Physica* 272: 173-187.

Batagelj, V. and Mrvar, A: Pajek. (1998). A Program for Large Network Analysis. *Connections*, 21(2), 47-57.

Behrens, Teresa R. Gray Denis O. (2001). Unintended consequences of cooperative research: impact of industry sponsorship on climate for academic freedome and other graduate student outcome, *Research Policy* 30: 179-199.

Bessen, James and Hunt, Robert M. (2003). An Empirical Look at Software Patents.

Brennen, Alex V. (2003). GnuPG Keysigning Party HOWTO.

Coriat, Benjamin and Fabienne Orsi. (2002). Establishing a new intellectual property rights regime in the United States, *Research Policy* 31: 1491-1507.

Dalle, Jean-Michel and David, Paul A. (2003). The Allocation of Software Development Resources in 'Open Source' Mode.

Dasgupta, Partha and Paul A. David. (1994). Toward a new economics of science, *Research Policy* 23: 487-521.

David, Paul A. (2000). The Digital Technology Boomerang: New Intellectual Property Rights Threaten Global "Open Science", *forthcoming in World Bank Conference Volume, ABCD-2000.*

David, Paul A. (2003). The Economic Logic of "Open Science" and the Balance between Private Property Rights and the Public Domain in Scientific Data and Information: A Primer. Stanford Institute for Economic Policy Research Discussion Paper No. 02-30.

David, Paul A. and Dominque Foray. (2003). Economic Fundamentals of the Knowledge Society, *Policy Futures in Education* .

Fleming L, Waguespack D. (2003) Merit, Status, or Networking? The Emergence of Leaders in a Technological Community. Harvard Business School Working Paper, Aug 28, 2003.

Freeman, Linton C. (1979).Centrality in Social Networks Conceptual Clarification , *Social Networks* 1 (3): 215-239.

Gallivan, M. J. 2001. "Striking a Balance between Trust and Control in a Virtual

Organization: A Content Analysis of Open Source Software Case Studies," Information Systems Journal (11): 277-304.

Heller, Michael A. and Rebecca Eisenberg. (1998). Can Patents Deter Innovation? The Anticommons in Biomedical Research, *Science* 280: 698-701.

Jaffe, Adam B. (2000). The U.S. patent system in transition: policy innovation and the innovation process, *Research Policy* 29: 531-557.

Jarvenpaa, S. and Leidner, D. 1999. "Communication and Trust in Global Virtual Teams," Organization Science (10): 791-815.

S. Koch and G. Schneider. (2000). *Results from Software Engineering Research into Open Source Development Projects Using Public Data*.. Located at http://opensource.mit.edu/papers/kochossoftwareeng ineering.pdf.

Kogut, Bruce and Anca Metiu. (2001). Open-Source Software Development and Distributed Innocation, *Oxford Review of Economic Policy* 17 (2): 248-264.

Lessig, Lawrence. (2002). *The Future of Ideas: The Fate of the Commons in a Connected World.* New York: Vintage Books.

Long, J. S. (1997). Regression Models for Categorical and Limited Dependent Variables. Thousand Oaks, CA: Sage.

Marsden, Peter V. (1982). Brokerage Behavior in Restricted Exchange Networks. *Social Structure and Network Analysis*, Peter V. Marsden and Nan Lin. Beverly Hills: Sage Publications.

Merton, Robert K. (1968). The Matthew Effect in Science, *Science* 159: 56-63.

Merton, Robert K. (1937). Science, Population and Society, *The Scientific Monthly* 44 (2): 165-171.

Merton, Robert K. (1973). *The Sociology of Science.* Edited by Norman W. Storer. Chicago, IL: University of Chicago Press.

Michlmayer M, & Mako Hill B. "Quality and the Reliance on Individuals in Free Software Projects" Presented at *Taking Stock of the Bazaar: 3rd Workshop on OS SW Eng,* Portland OR, May 3-10, 2003. Located at: http://opensource.ucc.ie/icse2003/

Mockus A., Fielding, R.T, Herbsleb, J. (2000). *A Case Study of Open Source Software Development: The Apache Server*, Proceedings of ICSE'.

Mockus, Audris, Roy T. Fielding, and James D. Herbsleb. (2002). Two Case Studies of Open Source Software Development: Apache and Mozilla, *ACM Transactions on Software Engineering and Methodology* 11 (3): 309-346.

Mowery, David C. and Bhaven Sampat. (2004). The Bayh Dole Act of 1980 and University Industry Technology Transfer: A Model for Other OECD Governments? In *'Ivory Tower' and Industrial Innovation: University-Industry Technology Transfer Before and After the Bayh-Dole Act.* Palo Alto: Stanford University Press.

Mowery, David C. and Arvids A. Ziedonis. (2002). Academic patent quality and quantity before and after the Bayhl-Dole act in the United States, *Resarch Policy* 31 (3): 399.

Murdock, I. "A Brief History of Debian". Appendix A, The Debian Manifesto, Revised January 6, 1994, located at: http://www.debian.org/doc/manuals/project-history/ap-manifesto.en.html.

Network Associates. (2003). An Introduction to Cyptography..

Nohria, Nitin and Robert G. Eccles. (1992). Face-to-Face: Making Network Organizations Work. In *Networks and organizations structure, form, and action.* Edited by Nitin Nohria and Robert G Eccles. Boston, Mass: Harvard Business School Press.

O'Mahony, Siobhan. (2002). Community Managed Software Projects: The Emergence of a New Commercial Actor, unpublished dissertation, Stanford University.

O'Mahony, Siobhan. (2003). Guarding the Commons: how community managed projects protect their work, *Research Policy* (32): 1179-1198.

O'Mahony, Siobhan. (2004). Managing Community Software in a Commodity World, to appear in *Ethnographic Reflections on the New Economy*, (Eds.) Melissa Fisher, and Greg Downey, Duke University Publications.

Owen-Smith, Jason. (2003). From separate systems to a hybrid order: accumulative advantage across public and private science at Research One universities, *Research Policy* 32: 1081-1104.

Owen-Smith, Jason and Walter W. Powell. (2003). The expanding role of university patenting in the life sciences: assessing the importance of experience and connectivity, *Research Policy.*

Powell, Walter W. and Jason Owen-Smith. (1998). Universities and the market for intellectual property in the life sciences, *Journal of Policy Analysis and Management* 17 (2): 253-277.

Raymond, Eric S. (1999). *The Cathedral and the Bazaar: Musings on Linux and Open Source by an Accidental Revolutionary.* Sebastopol, CA: O'Reilly & Associates.

Scotchmer, Suzanne. (1999). Cumulative Innovation in Theory and Practice. Goldman School of Public Policy Working Paper 240, University of California (Berkeley).

Scotchmer, Suzanne. (1996). Protecting early innovators: should second-generation products be patentable? *RAND Jounrl of Economics* 27 (2): 322-331.

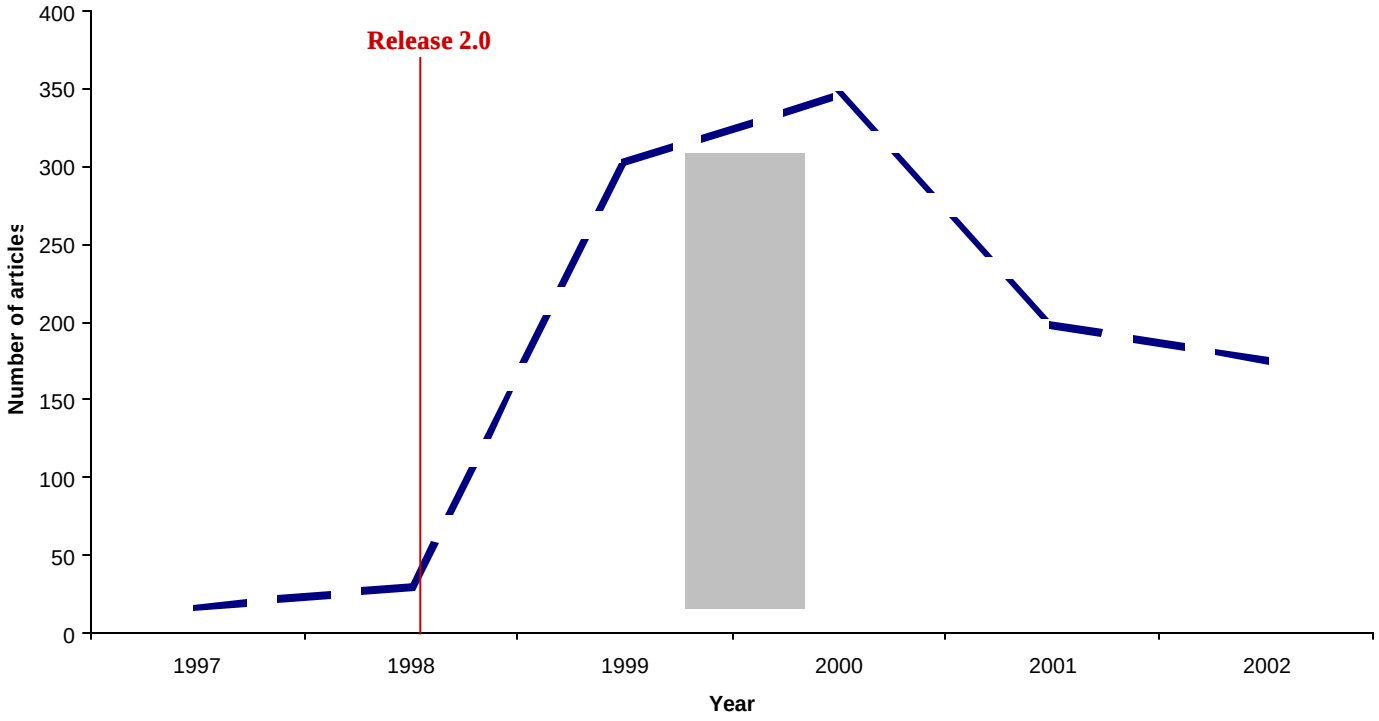
Scotchmer, Suzanne. (1991). Standing on the Shoulders of Giants: Cumulative Research and the Patent Law, *Journal of Economic Perspectives* 5 (1): 29-41.

Ulrich, Bill. Debian GNU/Linux 2.2 (Potato), *PC Magazine*, November 13, 2001, located at http://www.pcmag.com/article2/0,4149,16093,00.asp.

Von Krogh, Georg, Sebastian Spaeth, and Karim R. Lakhani. (2003). Community, Joining, and Specialization in Open Source Software Innovation: A Case Study, *Research Policy* forthcoming.

Wasserman, Stanley and Joseph Galaskiewicz. (1994). *Advances in social network analysis: Research in the social and behavioral sciences.* Thousand Oaks, Calif: Sage Publications.

Figure 1: Media Citations of Debian GNU/Linux 1997 - 2002



The shaded area indicates the time during which the Debian project closed its doors to new members.

Source: Data on articles citing "Debian Linux" was collected from ABI/Inform database in Proquest and the Factiva database in 2003.

Fig. 2 Degree Distribution of Debian Keyring Network, 1997-2002.

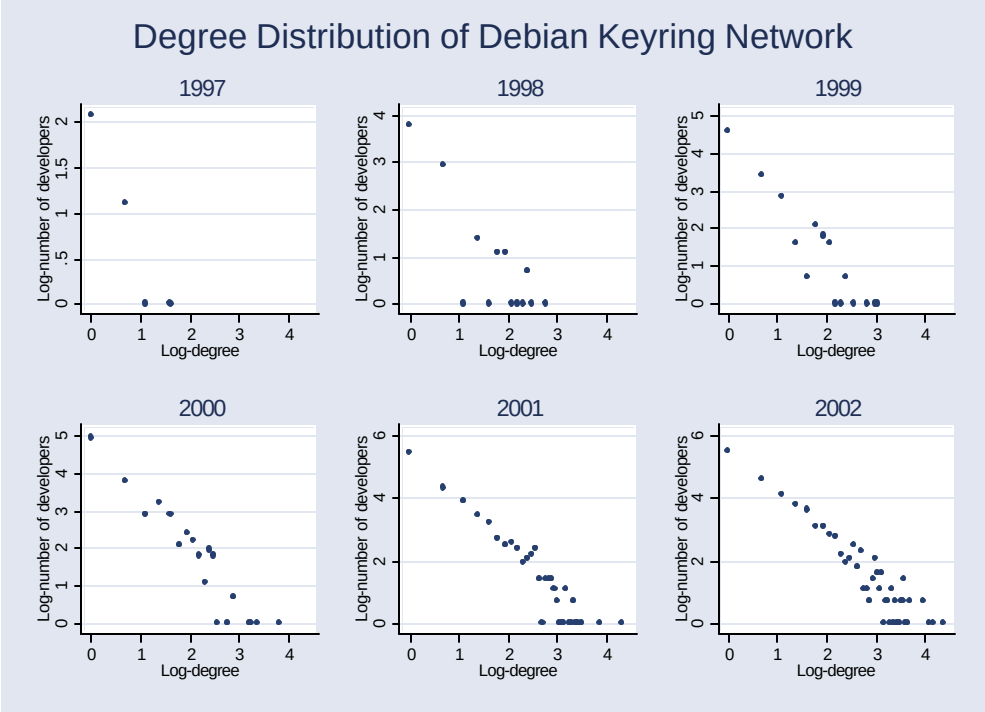
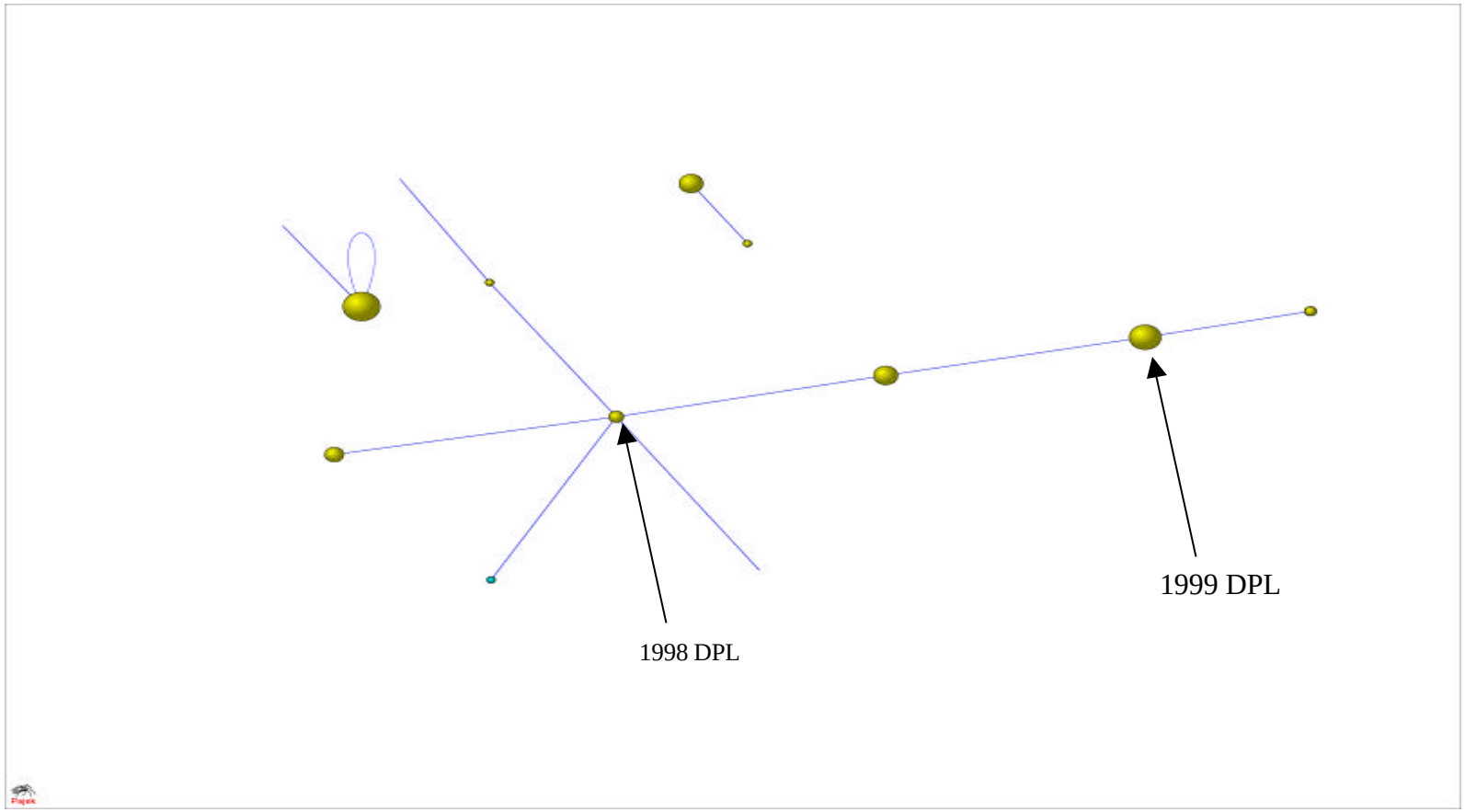


Figure 3: Debian Keyring Network, January 1, 1997 **Nodes = 13**



Note: In the social network visualization, green nodes indicate developers in the North America, gold in Europe, red in Asia, blue in South America, pink in Oceania, and turquoise for others. The size of the node measures the number of package maintained by the developer.

Figure 4: Debian Keyring Network, January 1,998 **Nodes = 82**

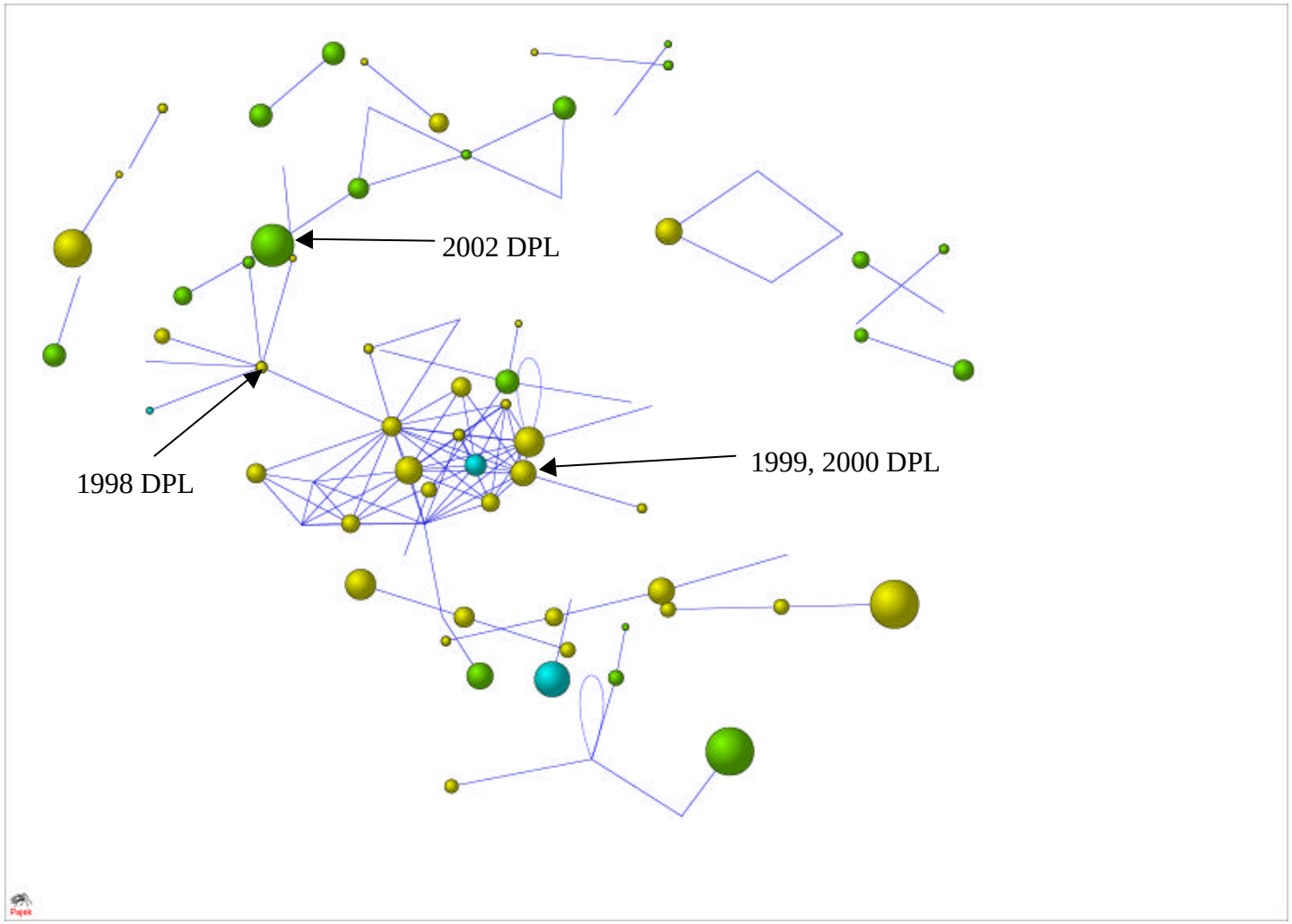


Figure 5: Debian Keyring Network, January 1, 1999 **Nodes = 176**

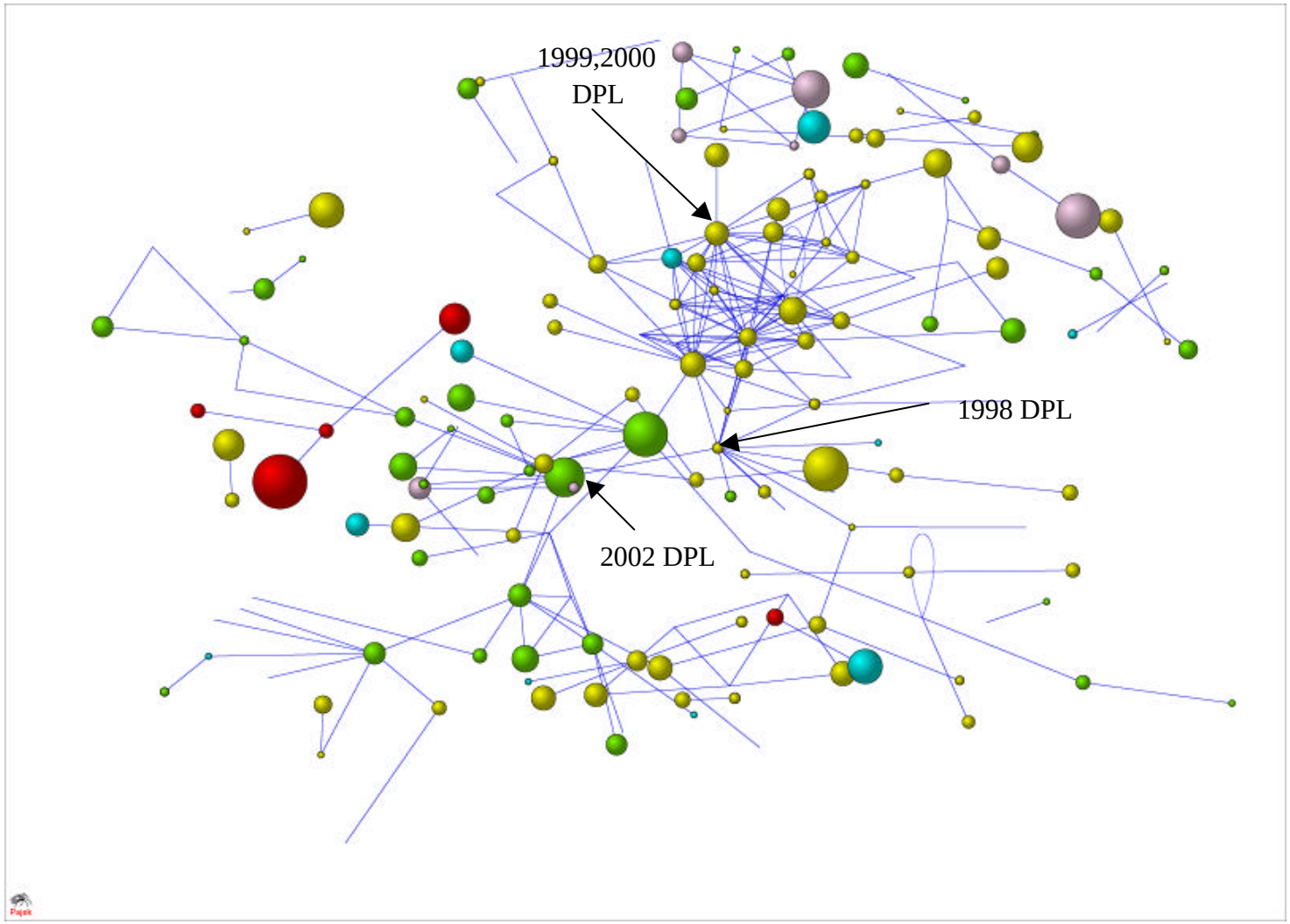


Figure 6: Debian Keyring Network: January 1, 2000 **Nodes = 298**

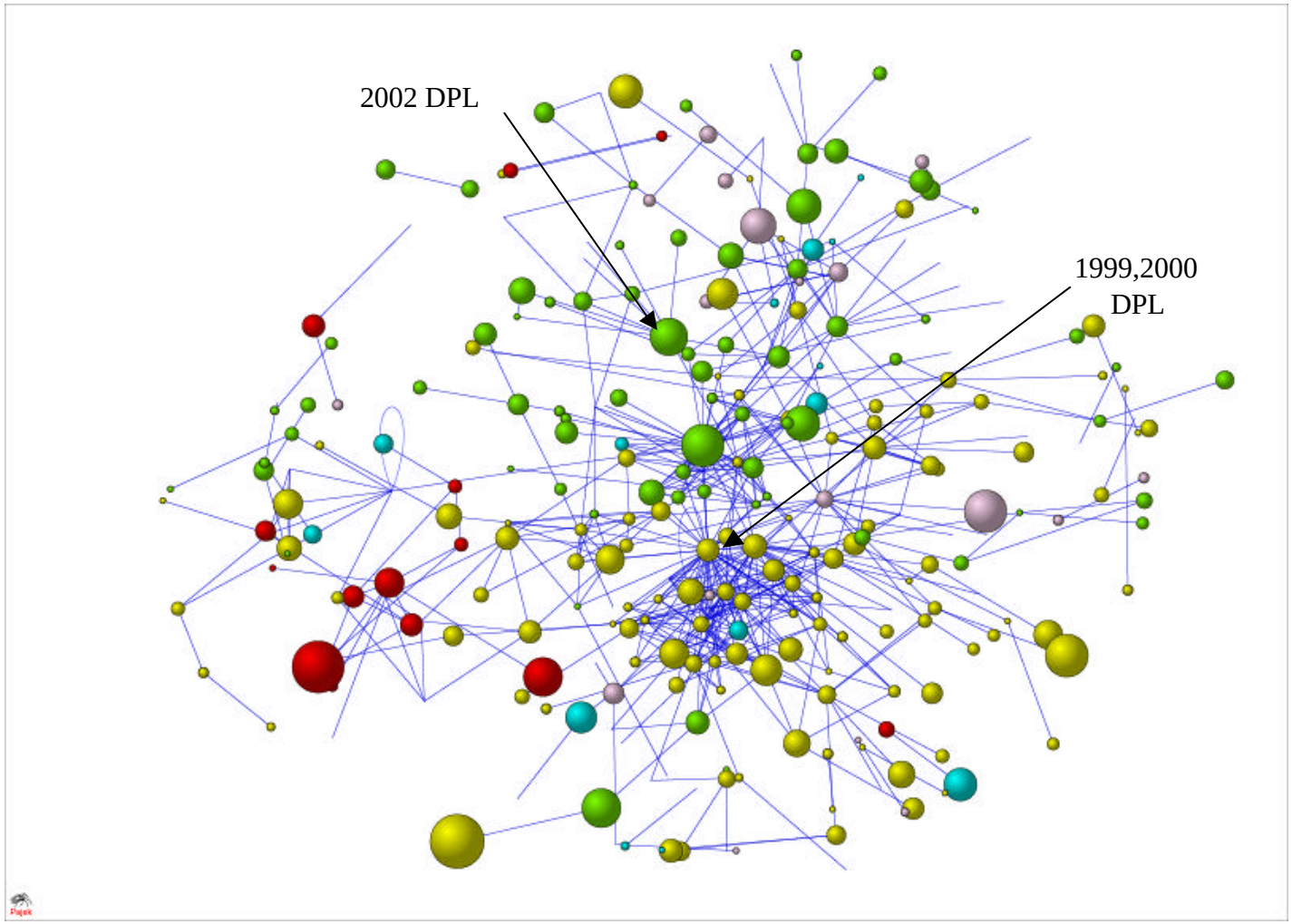


Figure 7: Debian Keyring Network: January 1, 2001 **Nodes = 532**

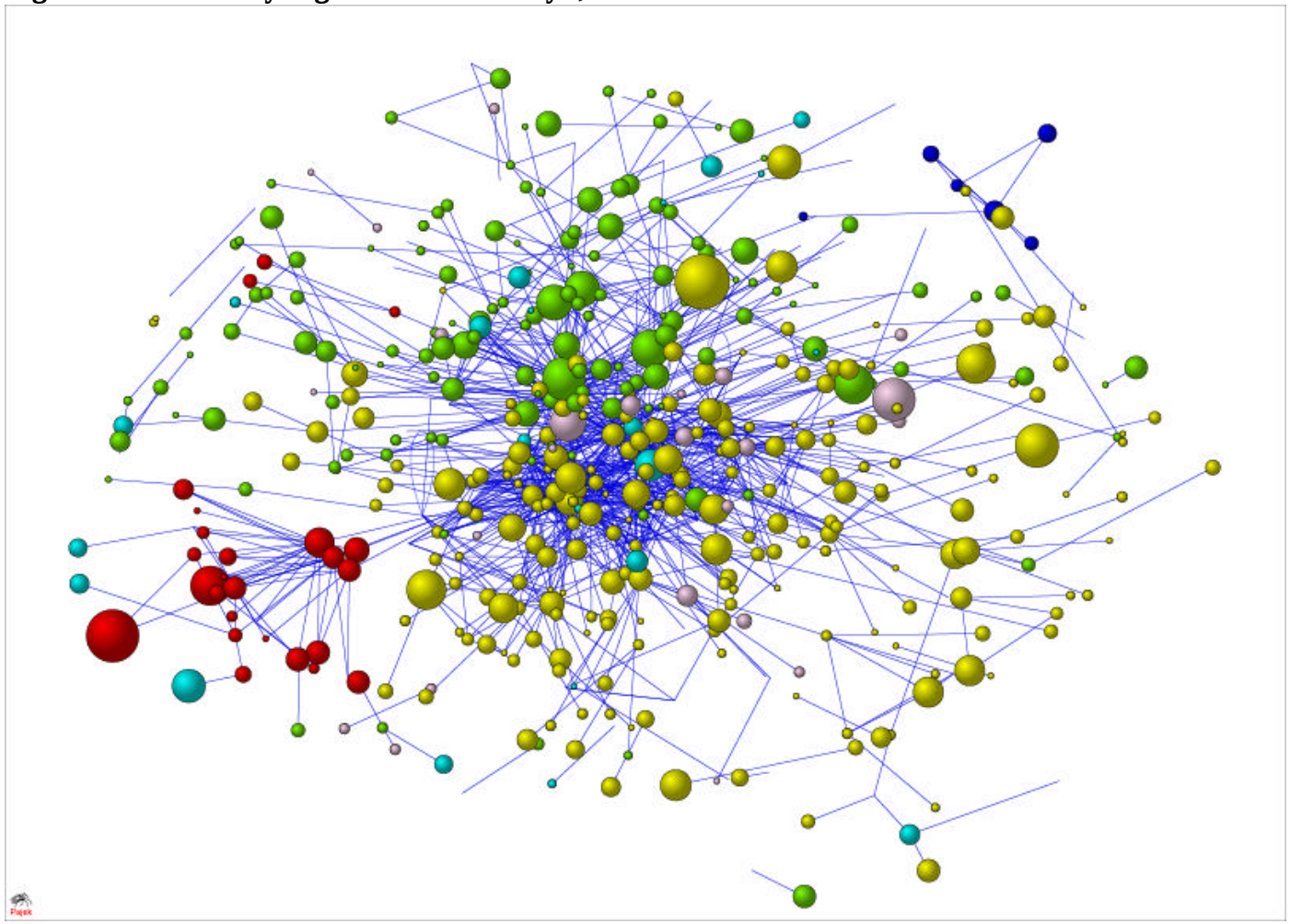
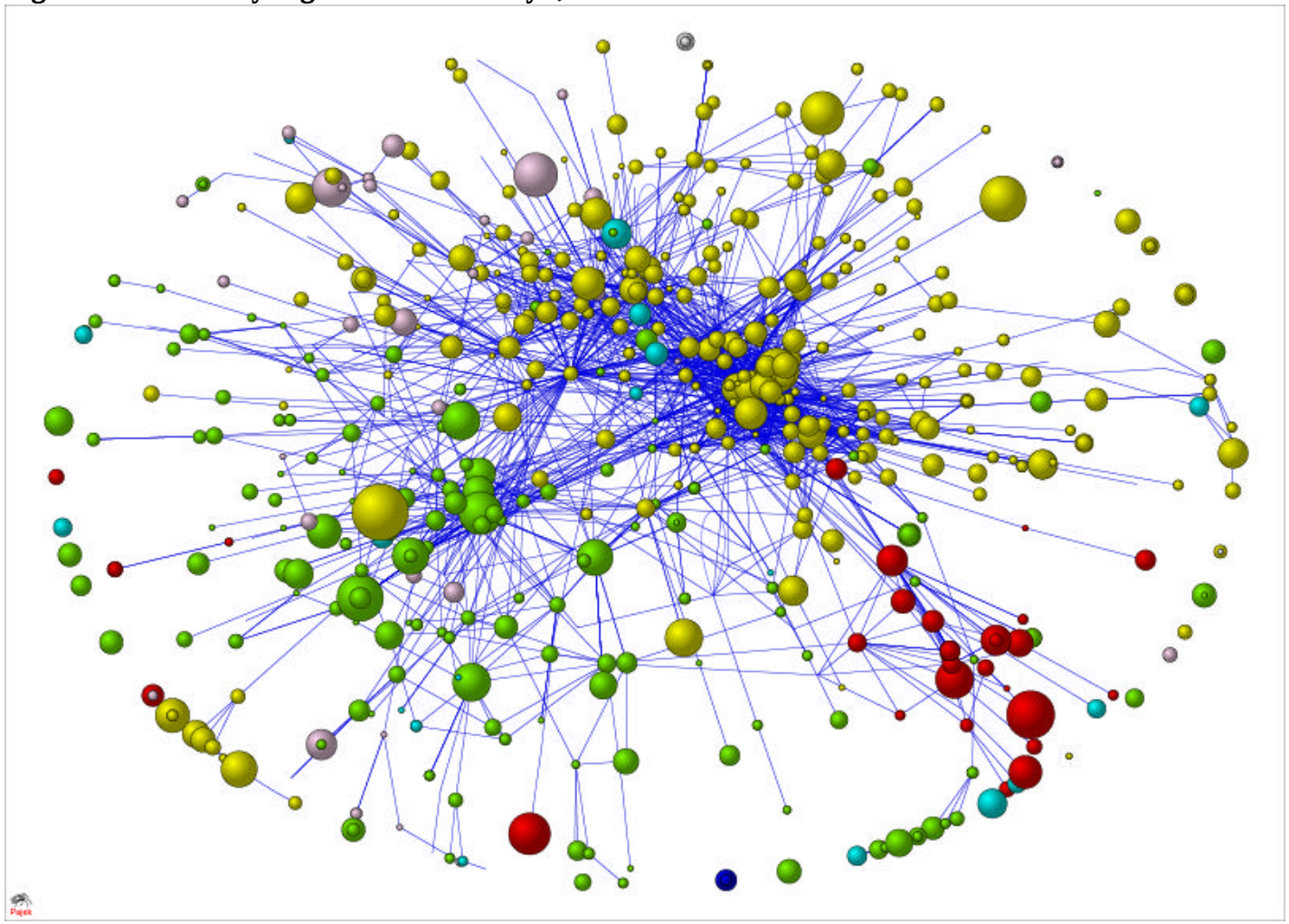
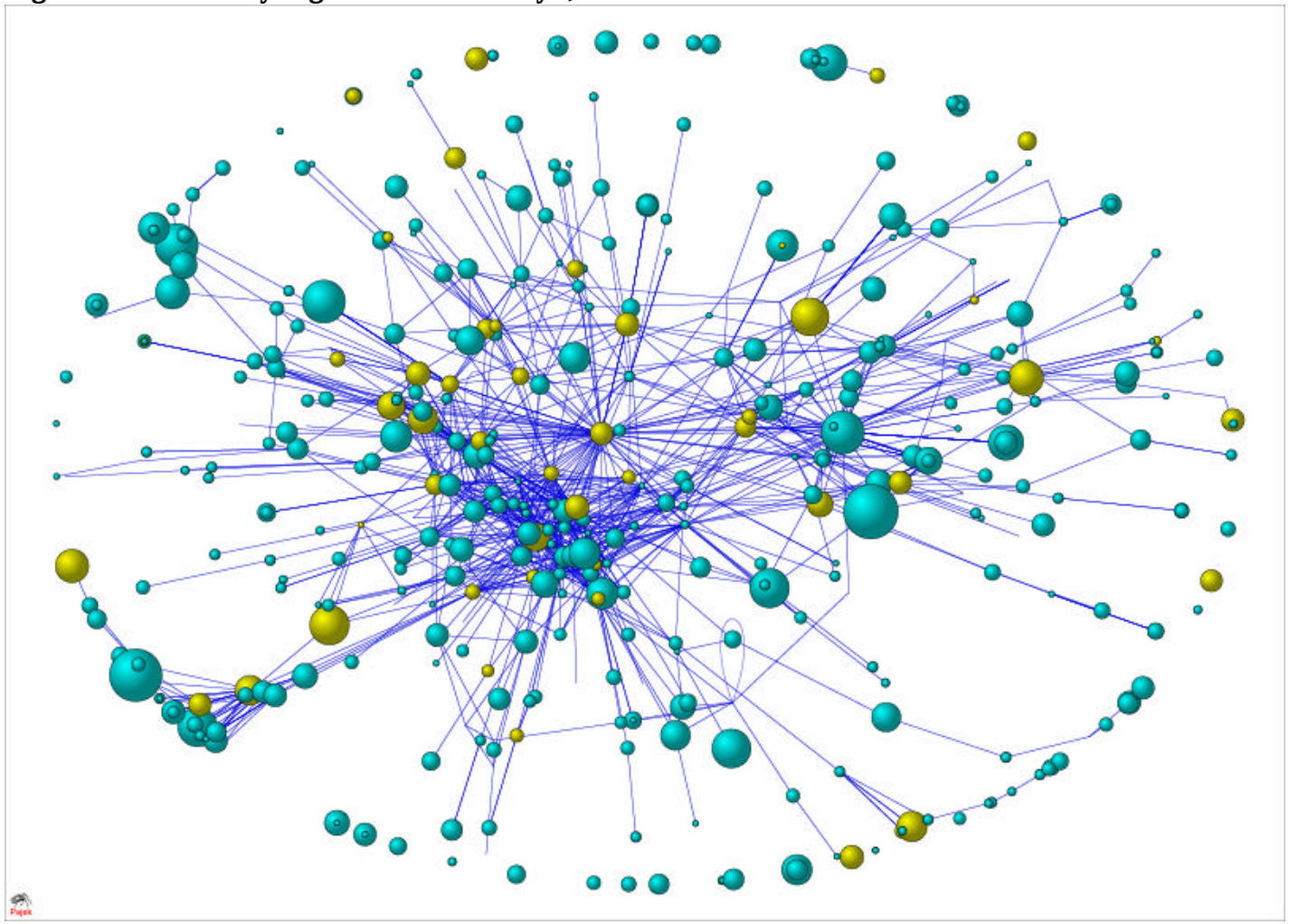


Figure 8: Debian Keyring Network: January 1, 2002 **Nodes = 671**



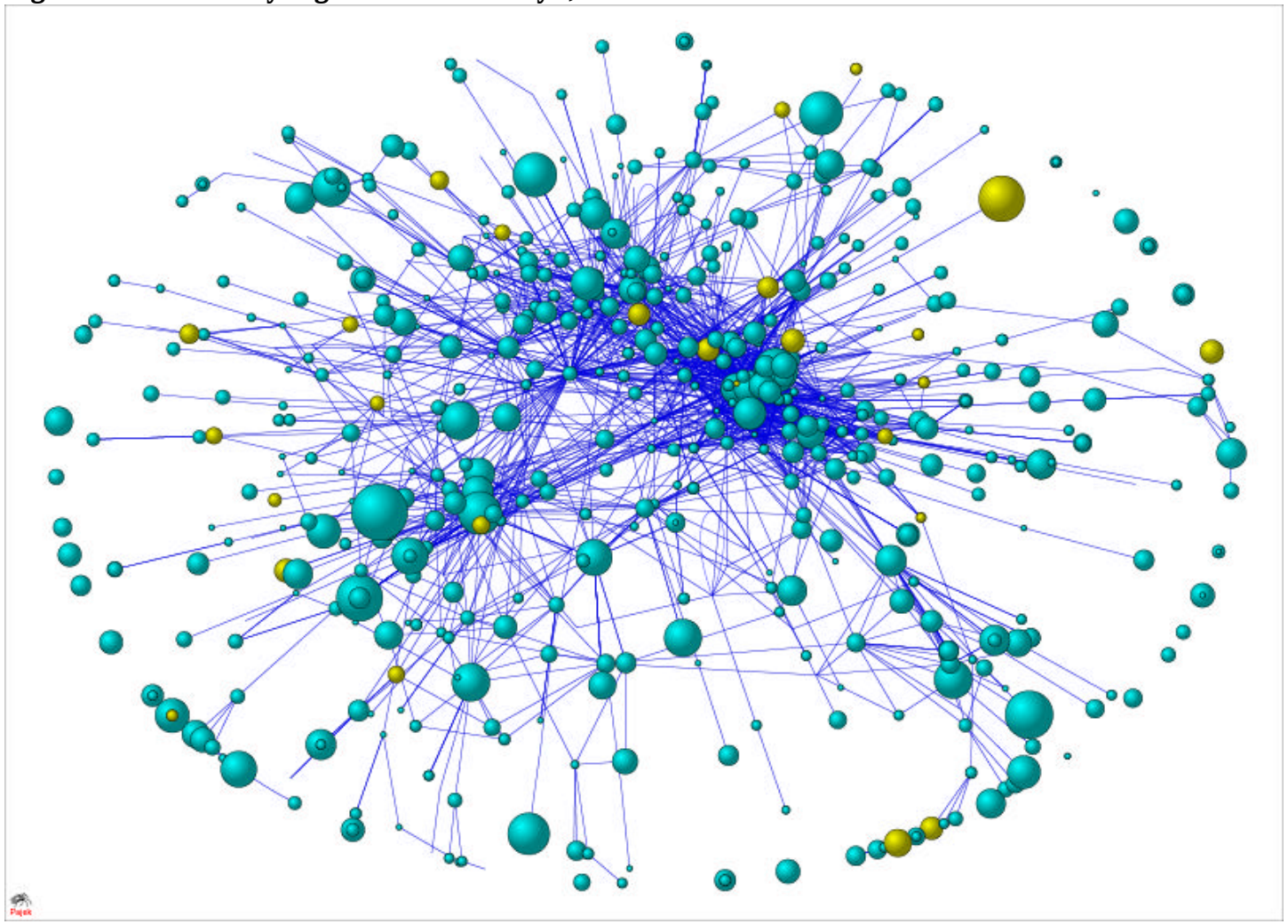
Note: In this social network visualization, green nodes indicate developers in the North America, gold in Europe, red in Asia, blue in South America, pink in Oceania, and turquoise for others. The size of the node measures the number of package maintained by the developer.

Figure 9: Debian Keyring Network: January 1, 2001 **Nodes = 532**



Note: In this social network visualization, the nodes in gold indicate developers who became members of the NMC. The size of the node measures the number of package maintained by the developer.

Figure 10: Debian Keyring Network: January 1, 2002 **Nodes = 671**



Note: In this social network visualization, the nodes in gold indicate developers who became members of the NMC. The size of the node measures the number of package maintained by the developer.

Table 1: Growth in the Debian Keyring Network

	1997	1998	1999	2000	2001	2002[a]
Number of Developers in network	13	82	176	298	532	671
Growth rate	-	530.77%	114.63%	69.32%	79.32%	26.13%
Number of ties	11	111	239	543	1212	2014
Average Degree	2.46	2.71	2.72	3.64	4.56	6.00
S.D. Degree	1.60	3.04	3.22	4.67	6.57	8.84
Number of Components (min 2)	3	19	29	31	39	33
Density	0.124	0.033	0.015	0.012	0.009	0.009

[a] Based on the developers' network on sep. 22, 2001

Table 2: Descriptive statistics of Debian Developers, 2001-2002.

	2001		2002	
	All Developers (N=532)		All Developers (N=671)	
	Mean	S.D.	Mean	S.D.
Dependent Variable				
New Maintaner Committee	.010		.005	
Independent Variables				
Number of Packages maintained	6.68	9.01	7.23	9.87
Package popularity	299.5	809.8	270.9	749.6
Tenure (in months)	18.54	15.50	22.89	16.41
Europe (reference category)[a]	.46		.47	
North-America[a]	.31		.31	
Other continent[a]	.12		.13	
Betweenness Centrality (*100)	.39	1.30	.37	1.04

[a]Dummy variables

Table 3: Logistic Regression coefficients for the regression of New Maintainer Committee membership on selected independent variables in 2001-2002

Independent Variables	**2001**		**2002**	
	(1)	**(2)**	**(1)**	**(2)**
Number of packages maintained	.03**	.04**	.02	.02*
		(1.04)		(1.03)
Package Popularity	.0003*	.0003	.0004**	.0004**
		(1.0003)		(1.0004)
Tenure in the project (in months)	-.02*	-.05***	-.06***	-.07***
		(.95)		(.93)
North America[a]	-.17	.03	.38	.46
		(1.03)		(1.58)
Other Continent[a]	-.35	-1.52	-.83	-.75
		(.86)		(.47)
Betweenness centrality		1.099***		.38***
		(2.99)		(1.47)
Intercept	-2.14***	-2.28***	-2.44***	-2.45***
Log-likelihood ratio for model estimated:				
vs. null model	13.96†	58.81††	20.36††	27.89††
(df)	(5)	(6)	(5)	(6)
vs. previous model		44.85††		7.52††
(df)		(1)		(1)
Pseudo R^2	0.04	0.18	0.08	.12
N	513	513	645	645

[a] Compared to developers located in Europe
Odds Ratios in parentheses
*=p<0.l, **=p<0.05, ***=p<0.01 (one tailed tests)
† χ^2 significant at the level (p = <.005)
†† χ^2 significant at the level (p = <.01)